関数の立体図の回転

２変数 x と y の関数

$$z = f(x, y)$$

の様子は曲面の立体図として視覚的に表すことができる。

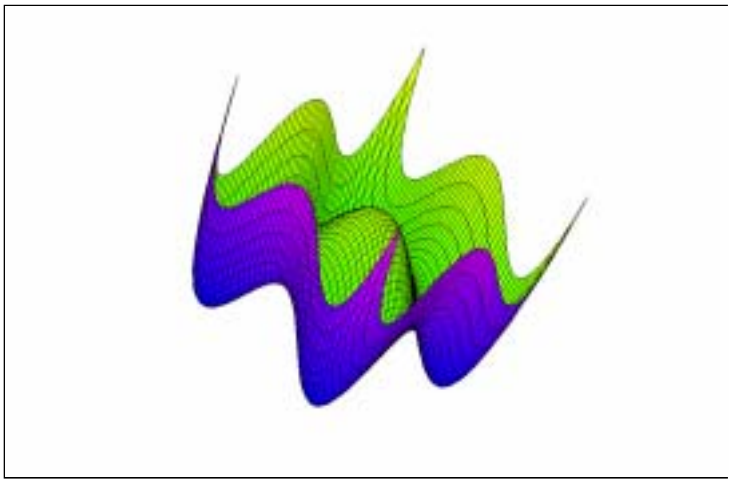

図１　関数の立体図

図１は関数

$$z = f(x, y) = \left(\frac{x}{2}\right)^4 + \left(\frac{y}{2}\right)^4 - 2\left\{\left(\frac{x}{2}\right)^2 + \left(\frac{y}{2}\right)^2\right\}$$

の $[-3 \quad 3]\times[-3 \quad 3]$ における関数値を表す曲面を描いたものである。図１では関数値の曲面を立方体の内部に描き、それを適当に回転して Y 軸の負の方向から見た状態で描いている（図２）。

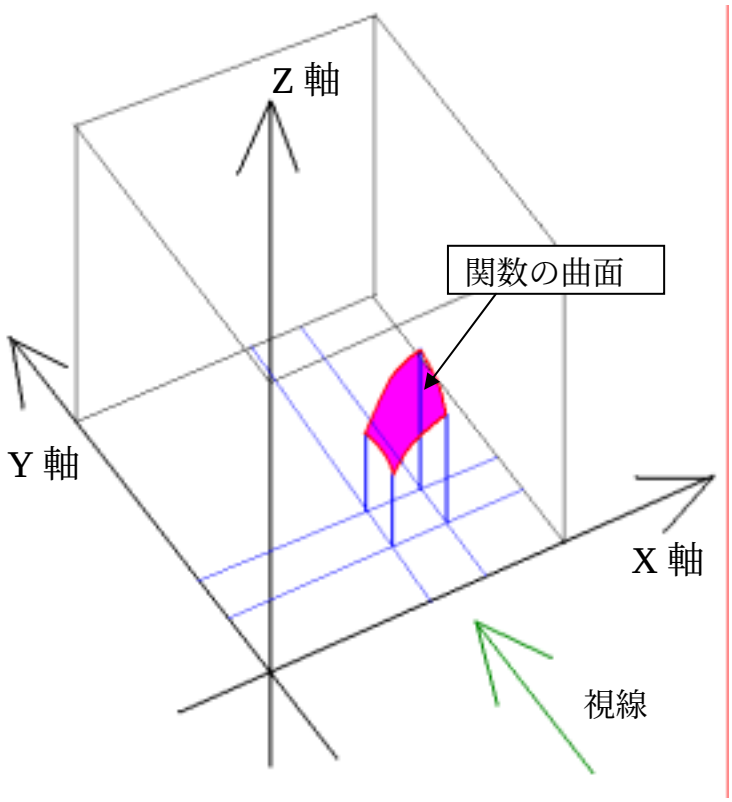

図２　関数の曲面と視線

関数の変域と関数値の範囲を適当に線形変換して立方体内にちょうど収まるようにした後、Y 軸の負の方向から見た点$(x,\ f(x,y))$の軌跡（曲面）を描いている。関数値は、変域を格子状に区切り、格子点(x_i, y_i)における関数値を求め、関数値の曲面を、格子点での値を通る、点$(x_i, y_i, f(x_i, y_i))$を頂点とする四角形を連結した面（タイルをつないだような面）で表している。

立方体内の関数の曲面は、オイラーの角（斉藤、1966）を設定して自由に回転することができる。オイラーの角とは、次のように３次元空間での回転を表す３つの角、θ、ϕ、ψ、のことである。Y 軸の回りの角 θ の回転をY_θ、Z 軸の回りの角ϕおよび角ψの回転をZ_ϕおよびZ_ψで表わすと、回転 T は３つの回転の積

$$T = Z_\phi Y_\theta Z_\psi$$

として書き表すことができる。すなわち、

$$T = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad (1)$$

と書ける。（1）式による回転では、まず角ψによりZ軸の周りの回転が行われ、その後Y軸の回りの角θの回転が行われる。最後にこれら2つの回転後の新しいZ軸の周りでの角ϕでの回転が行われる。

回転後の描画は、視線をY軸の負の方向から正の方向に取り、Y軸の正の方向の後ろの曲面を表す四辺形（のタイル）から順番に前の方（Y軸の負の方向の方）のタイルを描いていくことにより、視線の前方の曲面から隠れている後ろの曲面の描画の処理を行っている。

プログラムの使い方

プログラムPRotPerspec.dprを実行すると先ず図？のフォームが表示される。

図3　実行開始時のフォーム

「GO」ボタンのクリックで図4の描画用フォームが表示され、回転を行わない状態での関数の曲面の描画が行われる。

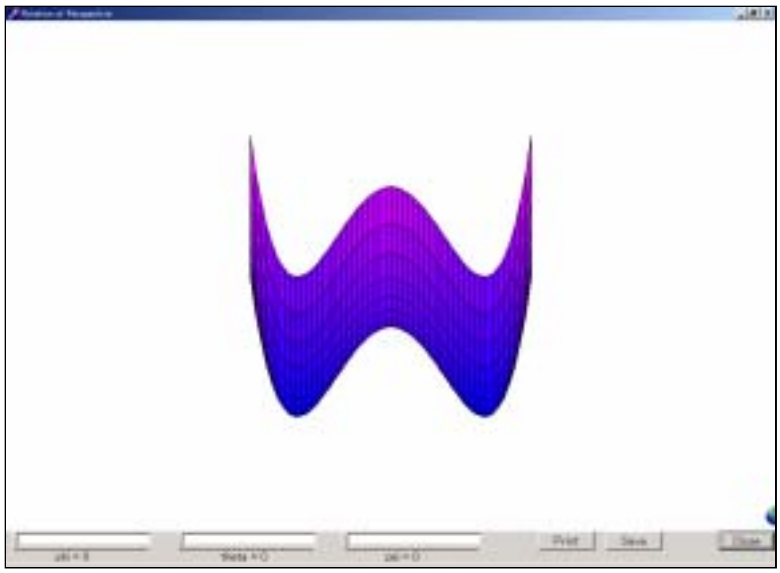

図４　回転前の描画

画面下に並んでいる３つのトラックバーを動かして適当に回転を行う。トラックバーを動かすとそれに合わせて回転された曲面の描画が表示される（図５）。

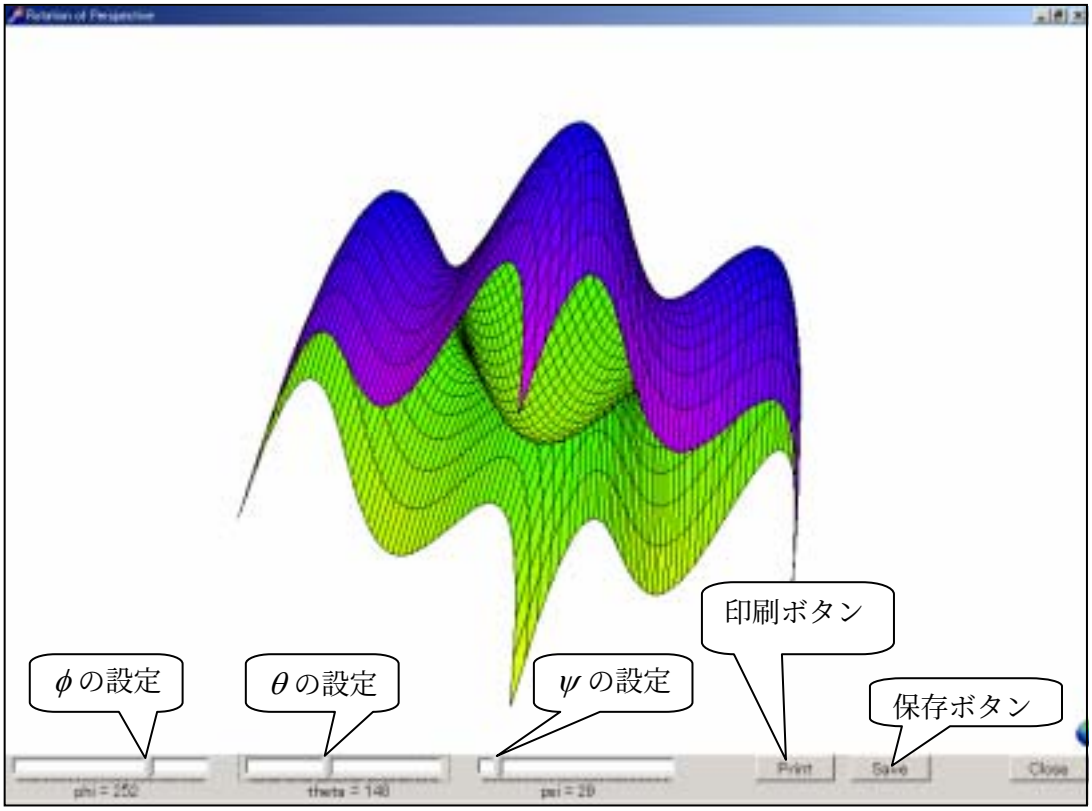

図５　回転

トラックバーの下には回転角が度を単位として表示されている。

表示されている図は、「Save」ボタンのクリックでファイルに保存することができる。「Print」ボタンをクリックすると描画図形がプリンタに出力される。

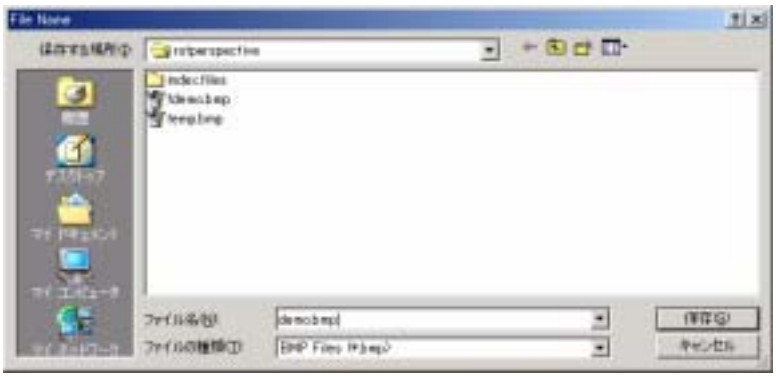

図６　画像保存用ファイル名の設定

「Save」ボタンをクリックすると図６のダイアログボックスが表示される。ファイルはビットマップファイル（*.bmp）として保存されるので、ファイル名は拡張子として.bmp を付けておく。ファイル名の設定後、「保存」ボタンをクリックすると設定した名前のファイルに描画図形が保存される。

図７　プリンタ用のダイアログボックス

「Print」ボタンをクリックすると図７のダイアログボックスが表示される。必要ならばプロパティボタンをクリックして適当な設定を行う。「OK」ボタンをクリックするとプリンタへの出力が始まる。

図５の「Close」ボタンのクリックでプログラムの実行が終了する。

描画用関数の設定

描画の対象となる関数は手続き DrawPerspec

```
procedure DrawPerspec( CheckF : TFunc;  //  立体図を描く関数
                       pxmin, pxmax,    //  ｘの変域（最小値と最大値）
                       pymin, pymax     //  ｙの変域（最小値と最大値）
                              : Extended );
```

のパラメタ CheckF に設定する。サンプルプログラム PRotPerspec.dpr では DrawPerspec がユニット URotPerspec.pas において呼び出されている。ユニット URotPerspec.pas は、リスト１のようになっている。

リスト１　手続き DrawPerspec の呼び出し

```
unit URotPerspec;        //  by Yasuharu Okamoto, JWU, 2003.2

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    GOButton: TButton;
    ExitButton: TButton;
    procedure ExitButtonClick(Sender: TObject);
    procedure GOButtonClick(Sender: TObject);
  private
    { Private 宣言 }
  public
    { Public 宣言 }
  end;

var
  Form1: TForm1;

type
  TFunc = function( x, y : Extended ) : Extended;  //  関数の型の宣言


implementation

{$R *.DFM}

uses
  UDispRotPerspec;      //  立体図を描くためのユニット

procedure TForm1.ExitButtonClick(Sender: TObject);
begin
            Close;
end;
```

```
function CheckF( x, y : Extended ) : Extended; //  立体図を描く曲面の関数
  begin
        CheckF:=sqr(sqr(0.5*x))+sqr(sqr(0.5*y))-2*(sqr(0.5*x)+sqr(0.5*y));
  end;

procedure TForm1.GOButtonClick(Sender: TObject);
begin
         ExitButton.SetFocus;  UpDate;

         //    立体図を描く手続きの呼び出し
         DrawPerspec( CheckF, -3.0, 3.0, -3.0, 3.0 );
end;

end.
```

フォーム上の「GO」ボタン（図３）のクリックで実行されるメソッド GOButtonClick において、手続き DrawPerspec が

```
DrawPerspec( CheckF, -3.0, 3.0, -3.0, 3.0 );
```

と呼び出されている。関数の変域は $\{(x,y)\mid -3\le x\le 3, -3\le y\le 3\}$ が指定されている。描画対象の関数として指定されている CheckF はメソッド GOButtonClick の前で

```
function CheckF( x, y : Extended ) : Extended;
  begin
        CheckF:=sqr(sqr(0.5*x))+sqr(sqr(0.5*y))-2*(sqr(0.5*x)+sqr(0.5*y));
  end;
```

と宣言されている。

また、ユニットの interface 部ではメソッド DrawPerspec の第１パラメタの型 TFunc が

```
type
  TFunc = function( x, y : Extended ) : Extended;
```

と宣言されている。Interface 部で宣言されているのは、この型名が DrawPerspec の宣言されているユニット UDispRotPerspec で使用されているからである。UDispRotPerspec の uses 節においては、ユニット URotPerspec の使用が宣言されている。URotPerspec では、DrawPerspec の宣言されているユニット UDispRotPerspec の使用が uses 節において宣言されている（図８）。

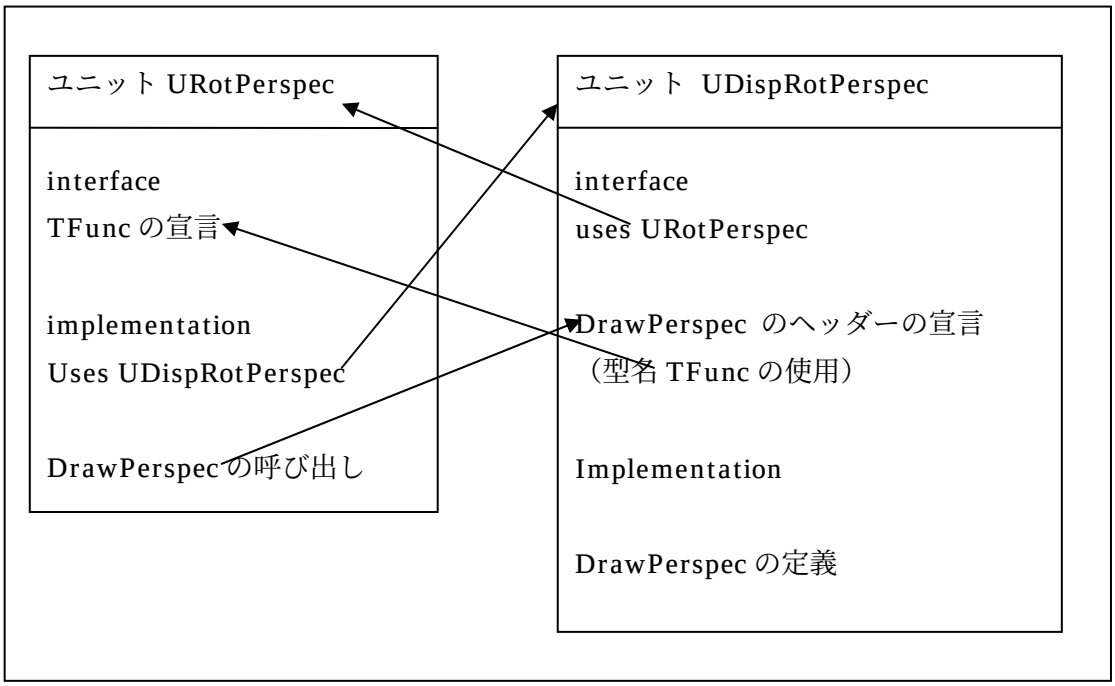

図８　ユニットの関係

参 考 文 献

（１）斎藤正彦「線型代数入門」、Pp.278、東京大学出版会、1966.